

FIG. 1 A

LS0 TYPE (BASE RELATIVE ADDRESSING)

opcode	dst	base	offset
--------	-----	------	--------

FIG. 1 B

LS1 TYPE (POST OFFSET UPDATE ADDRESSING)

opcode	dst	base	rptamt	offset
--------	-----	------	--------	--------

FIG. 1 C

LS2 TYPE (INDEX MODIFICATION/POST REGISTER UPDATE ADDRESSING)

opcode	dst	base	rptamt	extension	index
--------	-----	------	--------	-----------	-------

FIG. 2

extension	ADDRESS MODIFICATION METHOD
000	<p>INTERPRETS THE CONTENT OF THE REGISTER SPECIFIED IN THE "INDEX" AS SIGNED AND PERFORMS ADDRESS MODIFICATION.</p> <pre> for(t=0; t < rptamt; t++){ targetAdrs = base.uw + index[t].h[LOWER]; } </pre>
001	<p>INTERPRETS THE LOWER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" (<code>index[t].h[LOWER]</code>) AS SIGNED AND PERFORMS ADDRESS MODIFICATION.</p> <pre> for(t=0; t < rptamt; t++){ targetAdrs = base.uw + index[t].h[LOWER]; } </pre>
010	<p>INTERPRETS THE UPPER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" (<code>index[t].h[UPPER]</code>) AS SIGNED AND PERFORMS ADDRESS MODIFICATION.</p> <pre> for(t=0; t < rptamt; t++){ targetAdrs = base.uw + index[t].h[UPPER]; } </pre>
011	<p>INTERPRETS EACH OF THE LOWER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" (<code>index[t].h[LOWER]</code>) AND THE UPPER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" (<code>index[t].h[UPPER]</code>) AS SIGNED AND ALTERNATELY PERFORMS ADDRESS MODIFICATION.</p> <pre> for(t=0; t < rptamt; t++){ if ((t & 1)==0) targetAdrs = base.uw + index[t].h[LOWER]; else targetAdrs = base.uw + index[t].h[UPPER]; } </pre>

FIG. 3

extension	ADDRESS MODIFICATION METHOD
000	<p>INTERPRETS THE CONTENT OF THE REGISTER SPECIFIED IN THE "INDEX" AS SIGNED AND UPDATES THE ADDRESS.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; T += index[t].w; } base.uw = T; </pre>
001	<p>INTERPRETS THE LOWER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" , index[t].h[LOWER] AS SIGNED AND UPDATES THE ADDRESS.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; T += index[t].h[LOWER]; } base.uw = T; </pre>
010	<p>INTERPRETS THE UPPER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" , index[t].h[UPPER] AS SIGNED AND UPDATES THE ADDRESS.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; T += index[t].h[UPPER]; } base.uw = T; </pre>
011	<p>INTERPRETS EACH OF THE LOWER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" , index[t].h[LOWER], AND THE UPPER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" , index[t].h[UPPER], AS SIGNED AND ALTERNATELY UPDATES THE ADDRESSES.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; if ((t & 1) == 0) T += index[t].h[LOWER]; else T += index[t].h[UPPER]; } base.uw = T; </pre>
100	<p>INTERPRETS THE CONTENT OF THE REGISTER SPECIFIED IN THE "INDEX" AS SIGNED AND UPDATES THE ADDRESS. BASE REGISTER IS NOT UPDATED.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; T += index[t].w; } </pre>
101	<p>INTERPRETS THE LOWER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" , index[t].h[LOWER] AS SIGNED, AND UPDATES THE ADDRESS. BASE REGISTER IS NOT UPDATED.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; T += index[t].h[LOWER]; } </pre>
110	<p>INTERPRETS THE UPPER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" , index[t].h[UPPER] AS SIGNED, AND UPDATES THE ADDRESS. BASE REGISTER IS NOT UPDATED.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; T += index[t].h[UPPER]; } </pre>
111	<p>INTERPRETS EACH OF THE LOWER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" , index[t].h[LOWER], AND THE UPPER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" , index[t].h[UPPER], AS SIGNED AND ALTERNATELY UPDATES THE ADDRESSES. BASE REGISTER IS NOT UPDATED.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; if ((t & 1) == 0) T += index[t].h[LOWER]; else T += index[t].h[UPPER]; } </pre>

FIG. 4

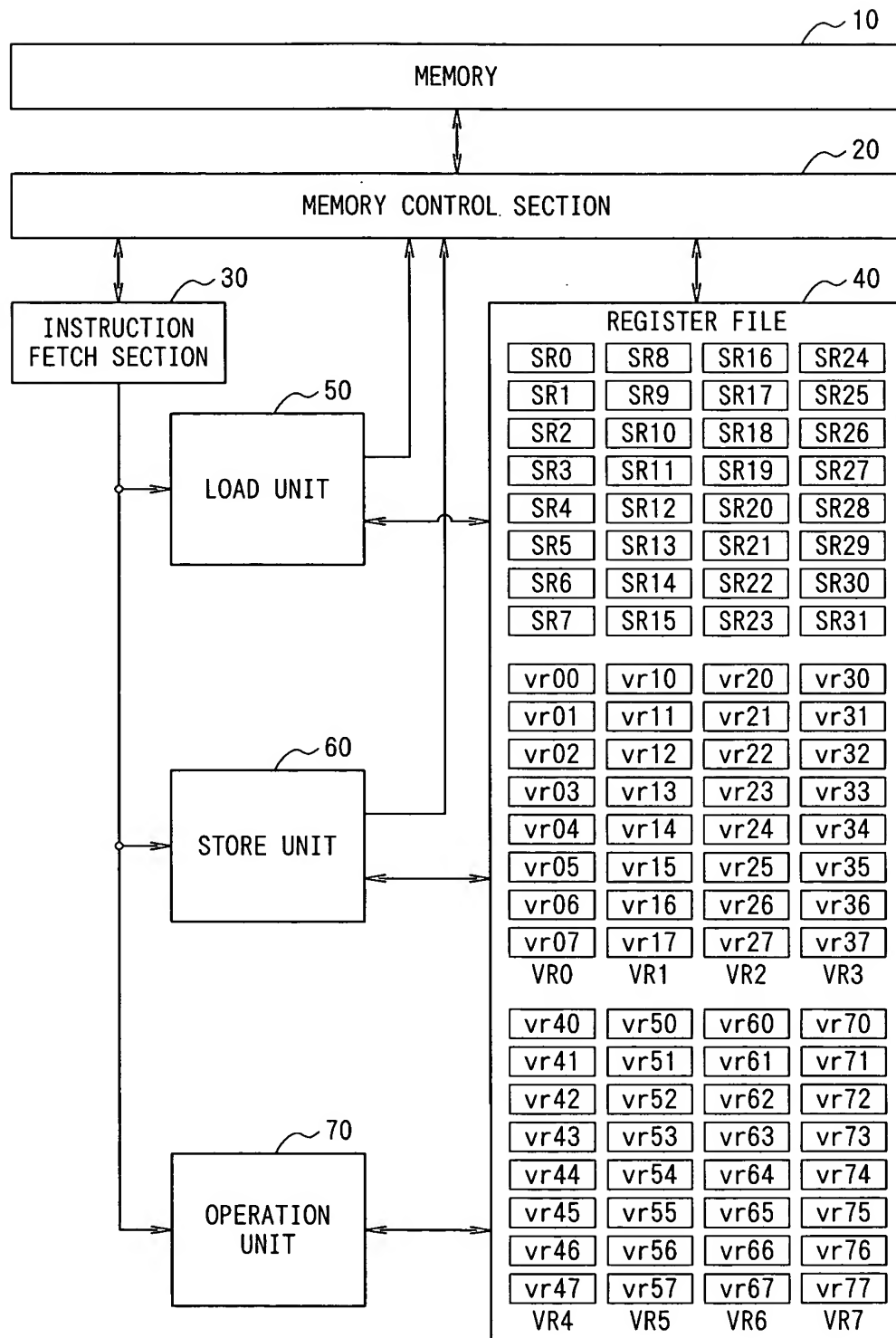


FIG. 5

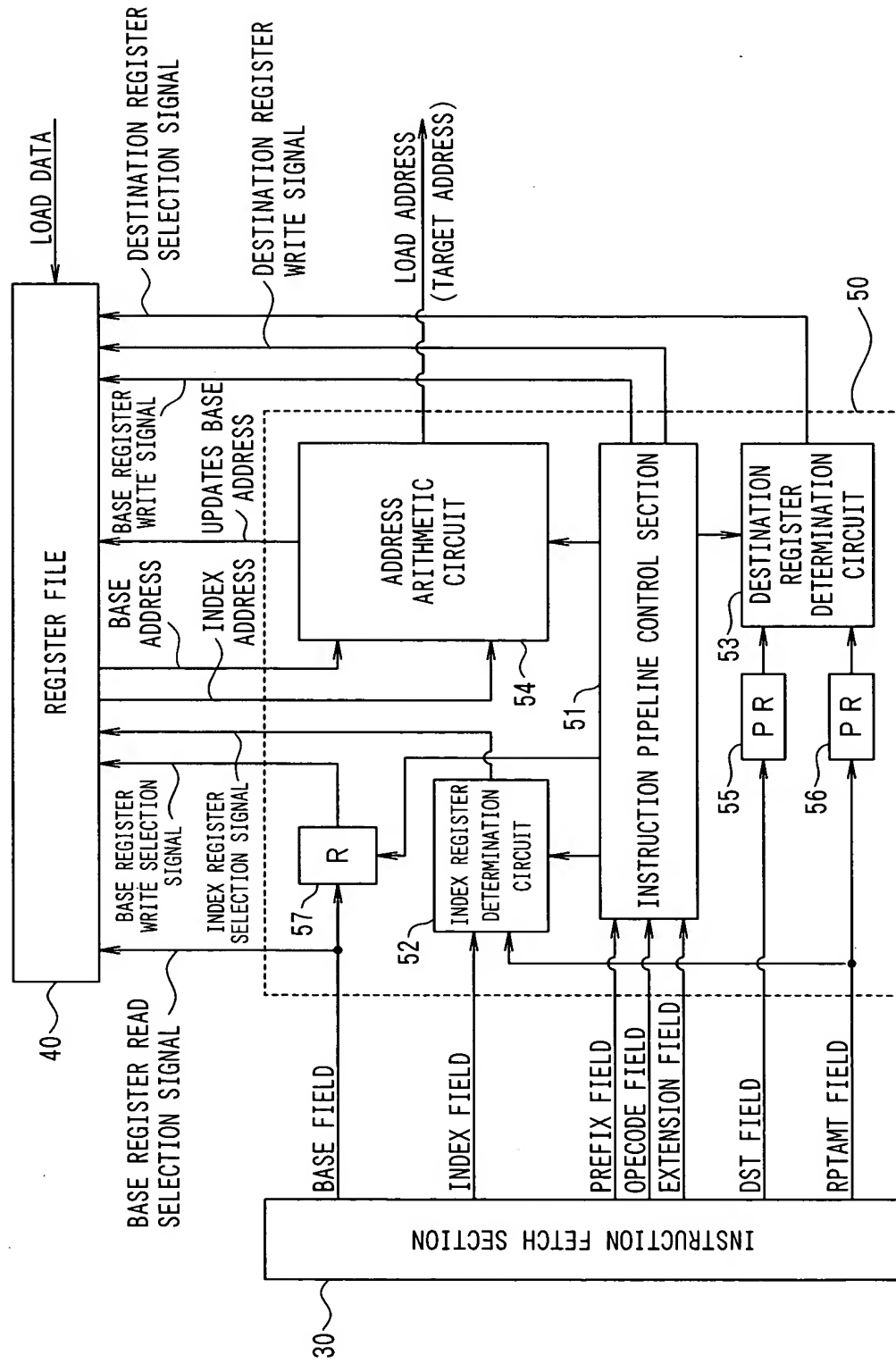


FIG. 6

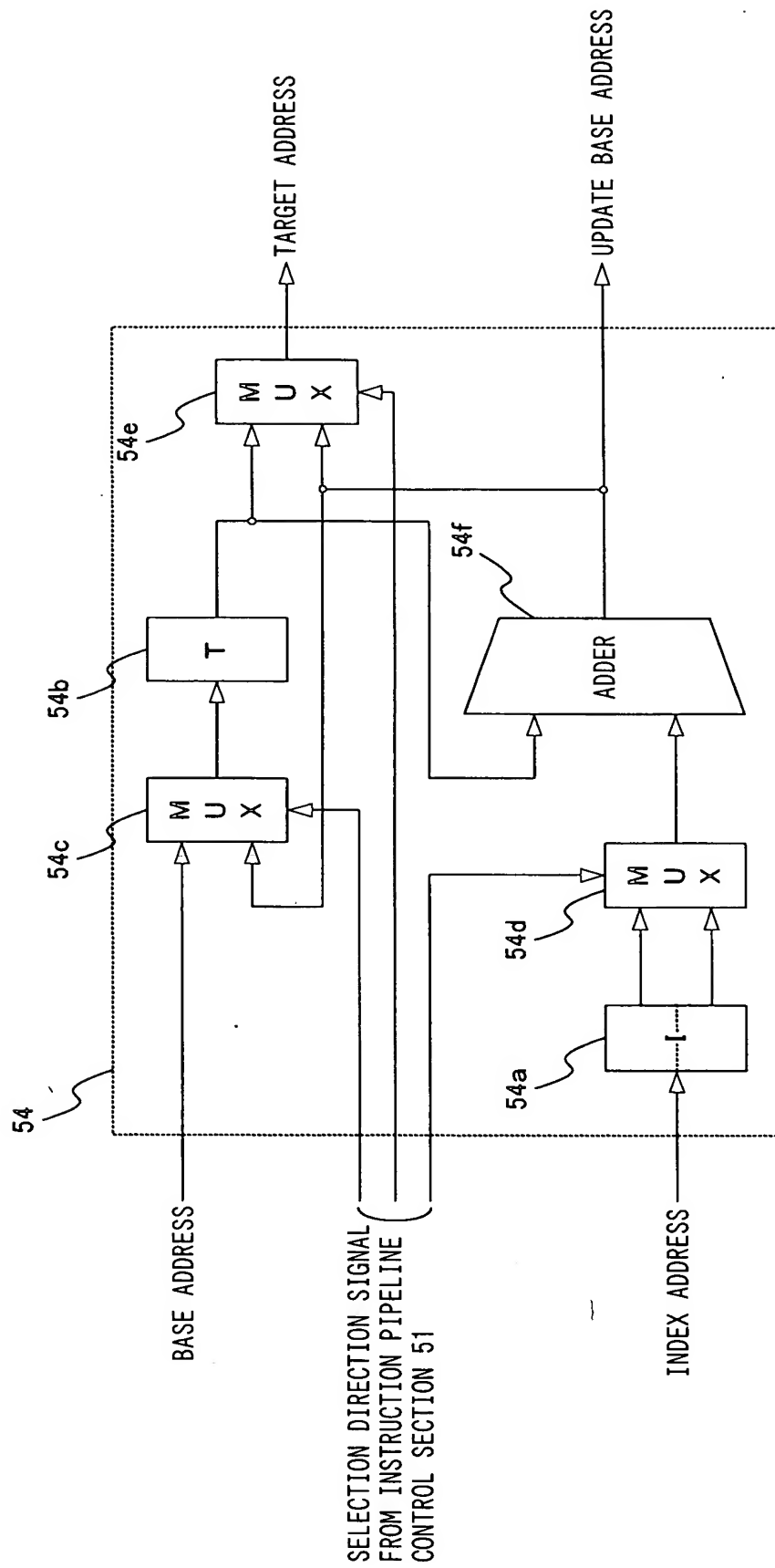


FIG. 7

extension	ADDRESS MODIFICATION METHOD
0000	<p>INTERPRETS THE CONTENT OF THE REGISTER SPECIFIED IN THE "INDEX" AS SIGNED AND PERFORMS ADDRESS MODIFICATION.</p> <pre> for(t=0; t < rptamt; t++){ targetAdrs = base.uw + index[t].w; } </pre>
0100	<p>INTERPRETS THE LOWER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" (index[t].h[LOWER]) AS SIGNED AND PERFORMS ADDRESS MODIFICATION.</p> <pre> for(t=0; t < rptamt; t++){ targetAdrs = base.uw + index[t].h[LOWER]; } </pre>
0110	<p>INTERPRETS THE UPPER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX" (index[t].h[UPPER]) AS SIGNED AND PERFORMS ADDRESS MODIFICATION.</p> <pre> for(t=0; t < rptamt; t++){ targetAdrs = base.uw + index[t].h[UPPER]; } </pre>

extension	ADDRESS MODIFICATION METHOD
0000	<p>INTERPRETS THE CONTENT OF THE REGISTER SPECIFIED IN THE "INDEX" AS SIGNED AND UPDATES THE ADDRESS.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; T += index[t].w; } base.uw = T; </pre>
0100	<p>INTERPRETS THE LOWER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX", index[t].h[LOWER] AS SIGNED, AND UPDATES THE ADDRESS.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; T += index[t].h[LOWER]; } base.uw = T; </pre>
0101	<p>THE SAME AS ABOVE. BASE REGISTER IS NOT UPDATED, HOWEVER.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; T += index[t].h[LOWER]; } </pre>
0110	<p>INTERPRETS THE UPPER 16 BITS OF THE REGISTER SPECIFIED IN THE "INDEX", index[t].h[UPPER] AS SIGNED, AND UPDATES THE ADDRESS.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; T += index[t].h[UPPER]; } base.uw = T; </pre>
0111	<p>THE SAME AS ABOVE. BASE REGISTER IS NOT UPDATED, HOWEVER.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; T += index[t].h[UPPER]; } </pre>
1100	<p>INTERPRETS ANY DATA OF THE LOWER 16 BITS index[t].h[LOWER] AND THE UPPER 16 BITS index[t].h[UPPER] OF THE REGISTER SPECIFIED IN THE "INDEX", AND THE LOWER 16 BITS SR30.h[LOWER] AND THE UPPER 16 BITS SR30.h[UPPER] OF THE REGISTER A AS SIGNED, AND ALTERNATELY UPDATES THE ADDRESSES.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; switch((ISW.h[LOWER] >> (2 * t)) & 0x3){ case 0: T += index.h[LOWER]; break; case 1: T += index.h[UPPER]; break; case 2: T += SR30.h[LOWER]; break; case 3: T += SR30.h[UPPER]; break; } } base.uw = T; </pre>
1101	<p>THE SAME AS ABOVE. BASE REGISTER IS NOT UPDATED, HOWEVER.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; switch((ISW.h[UPPER] >> (2 * t)) & 0x3){ case 0: T += index.h[LOWER]; break; case 1: T += index.h[UPPER]; break; case 2: T += SR30.h[LOWER]; break; case 3: T += SR30.h[UPPER]; break; } } </pre>
1110	<p>INTERPRETS ANY DATA OF THE LOWER 16 BITS index[t].h[LOWER] AND THE UPPER 16 BITS index[t].h[UPPER] OF THE REGISTER SPECIFIED IN THE "INDEX", AND THE LOWER 16 BITS SR31.h[LOWER] AND THE UPPER 16 BITS SR31.h[UPPER] OF THE REGISTER B AS SIGNED AND ALTERNATELY UPDATES THE ADDRESSES.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; switch((ISW.h[LOWER] >> (2 * t)) & 0x3){ case 0: T += index.h[LOWER]; break; case 1: T += index.h[UPPER]; break; case 2: T += SR31.h[LOWER]; break; case 3: T += SR31.h[UPPER]; break; } } base.uw = T; </pre>
1111	<p>THE SAME AS ABOVE. BASE REGISTER IS NOT UPDATED, HOWEVER.</p> <pre> T = base.uw; for(t=0; t < rptamt; t++){ targetAdrs = T; switch((ISW.h[UPPER] >> (2 * t)) & 0x3){ case 0: T += index.h[LOWER]; break; case 1: T += index.h[UPPER]; break; case 2: T += SR31.h[LOWER]; break; case 3: T += SR31.h[UPPER]; break; } } </pre>

FIG. 9

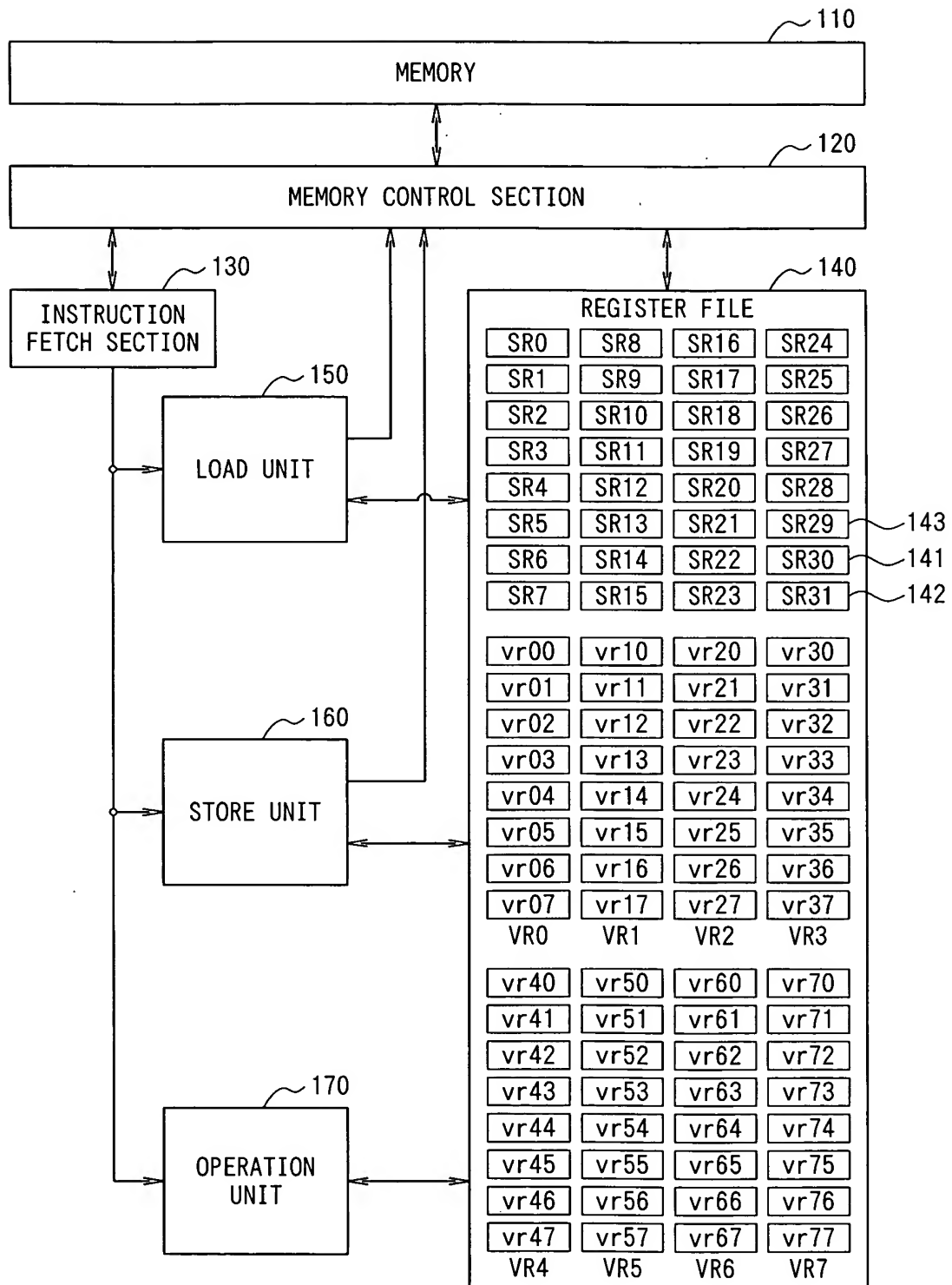


FIG. 10

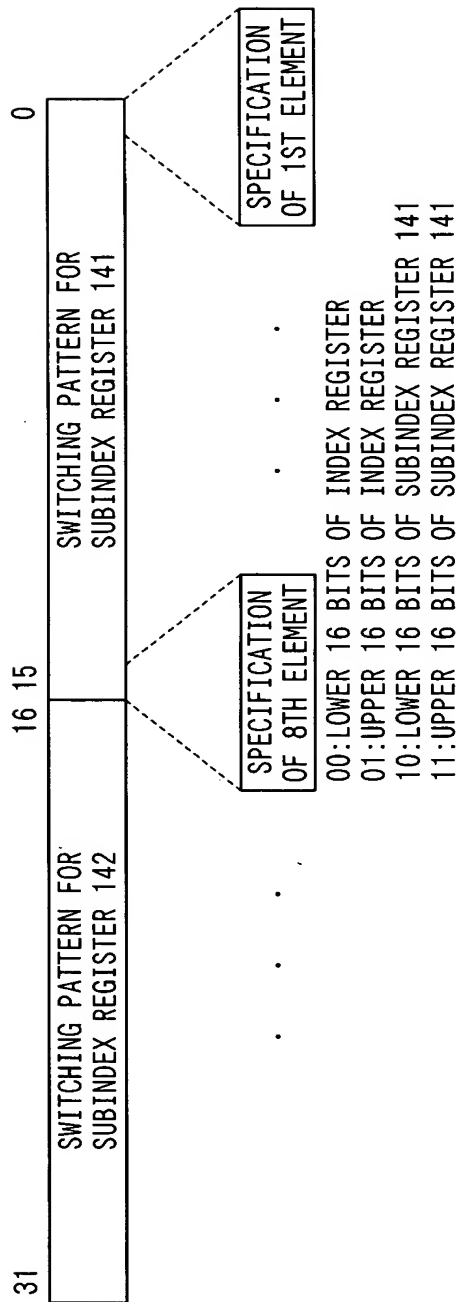


FIG. 12

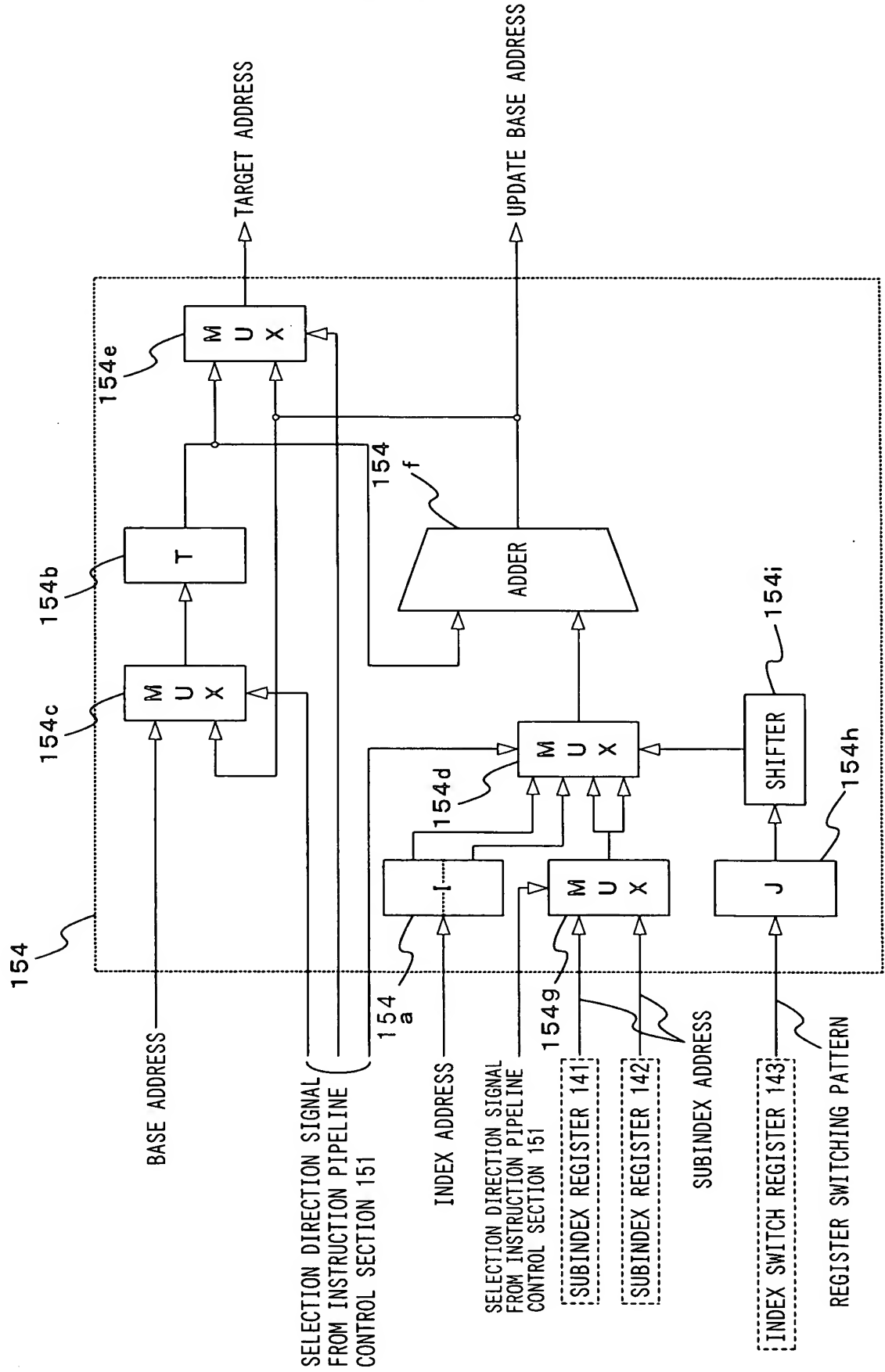


FIG. 13

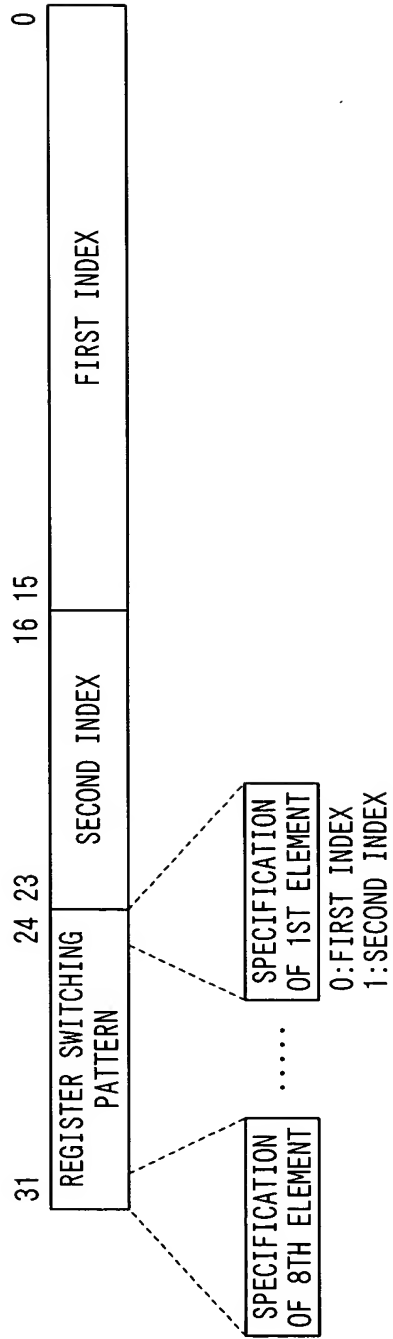


FIG. 14

